<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>ReVeaLD: A user-driven domain-specific interactive search platform for biomedical research</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author(s)</strong></td>
<td>Kamdar, Maulik; Hasnain, Ali; Decker, Stefan; Deus, Helena F.</td>
</tr>
<tr>
<td><strong>Publication Date</strong></td>
<td>2013</td>
</tr>
<tr>
<td><strong>Item record</strong></td>
<td><a href="http://hdl.handle.net/10379/4146">http://hdl.handle.net/10379/4146</a></td>
</tr>
</tbody>
</table>

Downloaded 2018-11-16T00:34:08Z

Some rights reserved. For more information, please see the item record link above.
ReVeaLD: A user-driven domain-specific interactive search platform for biomedical research

Maulik R. Kamdar a,*, Dimitris Zeginis b,c, Ali Hasnain a, Stefan Decker a and Helena F. Deus a

a Digital Enterprise Research Institute (DERI), National University of Ireland, Galway, Ireland
b Centre for Research and Technology Hellas, Thessaloniki, Greece
c Information Systems Lab, University of Macedonia, Thessaloniki, Greece

* Corresponding Author (Phone Number: +353 899529682)

E-mail Addresses: maulik.kamdar@deri.org (M. R. Kamdar), zeginis@iti.gr (D. Zeginis), ali.hasnain@deri.org (A. Hasnain), stefan.decker@deri.org (S. Decker), helena.deus@deri.org (H. F. Deus)
ABSTRACT

Bioinformatics research relies heavily on the ability to discover and correlate data from various sources. The specialization of life sciences over the past decade, coupled with an increasing number of biomedical datasets available through standardized interfaces, has created opportunities towards new methods in biomedical discovery. Despite the popularity of semantic web technologies in tackling the integrative bioinformatics challenge, there are many obstacles towards its usage by non-technical research audiences. In particular, the ability to fully exploit integrated information needs using improved interactive methods intuitive to the biomedical experts. In this report we present ReVeaLD (a Real-time Visual Explorer and Aggregator of Linked Data), a user-centered visual analytics platform devised to increase intuitive interaction with data from distributed sources. ReVeaLD facilitates query formulation using a domain-specific language (DSL) identified by biomedical experts and mapped to a self-updated catalogue of elements from external sources. ReVeaLD was implemented in a cancer research setting; queries included retrieving data from in silico experiments, protein modeling and gene expression. ReVeaLD was developed using Scalable Vector Graphics and JavaScript and a demo with explanatory video is available at http://srvgal78.deri.ie:8080/explorer. A set of user-defined graphic rules controls the display of information through media-rich user interfaces. Evaluation of ReVeaLD was carried out as a game: biomedical researchers were asked to assemble a set of 5 challenge questions and time and interactions with the platform were recorded. Preliminary results indicate that complex queries could be formulated under less than two minutes by unskilled researchers. The results also indicate that supporting the identification of the elements of a DSL significantly increased intuitiveness of the platform and usability of semantic web technologies by domain users.

Keywords – Domain-specific Languages; Human-Computer Interaction; Semantic Web Technologies; Knowledge Discovery; Visual Query System; Data Visualization
1 INTRODUCTION

The challenges in integrating the deluge of life sciences data flooding the web in the past decade have prompted researchers in bioinformatics and biomedical informatics to become early adopters of a new generation of integrative technologies based on Semantic Web and Linked Data concepts [1,2]. Various researchers have emphasized the use of these technologies for publishing biomedical data resources as SPARQL endpoints [3–7]. SPARQL [8] is the query language for the semantic web and the W3C-recommended standard for structured queries over highly heterogeneous information. These technologies have, on several occasions, facilitated naming disambiguation and retrieval of properties associated with bioinformatics-relevant entities (e.g. genes, proteins, drugs) from multiple sources [9–12]. Aggregated information gathered in this way can subsequently be applied to various bioinformatics tasks such as functional analysis, protein modeling or image analysis. In the majority of cases, the information necessary to answer a biological question requires data retrieval from various providers, each commonly making data available through custom interfaces. Questions such as “Are there drugs with molecular weight under 400 tested against ‘Colon Cancer’?” or “Do any Pubmed Publications refer to assays using ‘Aspirin’ as the primary Drug” can be accurately answered if information is aggregated in real-time from multiple reliable sources. Biomedical communities engaged in the Linked Data efforts have agreed extensively on ontologies and standards for exchanging experimental data and biomedical information [13,14]. Although these technologies have become commonplace for many computer scientists, who can easily interact with SPARQL endpoints and ontologies using dedicated applications, the ultimate end-users of the biomedical data exposed as SPARQL endpoints are primarily the domain experts - either biologists or clinical researchers, who can use it to improve their rate of discovery [15]. However, the assembly of SPARQL queries to aggregate information necessary for bioinformatics analysis still poses a high cognitive entry barrier, which continues to challenge wide adoption and acceptance of integrative technologies in biomedical domains. This problem is further compounded when we consider that most biomedical data sources are too large and dynamic for reliable data centralization [16], thus requiring federated SPARQL approaches [9,17]. Assembling a federated SPARQL query is a time-consuming and highly technical process,
even for computer scientists. It relies extensively on the familiarity with the data representation schema and the granularity of each federated source.

Visual query systems (VQSs) [18], powered by visual languages lower the cognitive entry level of assembling a SPARQL query, thus increasing the intuitiveness and usability of query languages [19]. Semantic visual query languages such as vSPARQL [20] and IML [21] support the assembly of queries with visual models of the data (typically the schema), guiding the user in the selection of the appropriate query elements. Freely available web-based query tools, such as Openlink iSPARQL [22] and NITELIGHT [23] provide graph-based, touch-and-click interfaces for query formulation where the outcome is a SPARQL query. These systems typically require a schema to be available at runtime or the schema is extracted by issuing SPARQL queries that extract the elements necessary to assemble queries [24]. These approaches are effective when targeting a single SPARQL endpoint but do not easily support federated SPARQL scenarios. Furthermore, we have shown in the past [25] that biomedical data representation schemas are still highly diverse, even when Linked Data technologies are applied. As such, even if those systems could be used in federated approaches, their reliance on schema extraction for assembling the visual models can easily overwhelm the users of the system due to too many overlapping terminologies, thus decreasing their intuitiveness [26]. Moreover, the display of the query results in these systems is still too geared towards textual representation; whereas faceted classification systems [27] like Exhibit [28] make the navigation of multi-dimensional datasets simpler and increase the presentation of results, the high-level of technical expertise required in configuring these interfaces makes them unfeasible for federated queries. In this report, we apply human-computer interaction (HCI) principles towards providing a methodology for the development of a federation-enabled semantic VQS and an optimized display of results.

1.1 HCI and Domain-specific Languages

In the context of HCI, usability describes the ease of use, intuitiveness and learnability of human-computer interactions [29]. One of the core principles in usability design is early and frequent focus on the users requirements and frequent interactions with early versions of the system - although this concept sounds intuitive, most semantic VQSs are not designed with this principle in mind and,
instead, focus more on extensively supporting every extension to the SPARQL protocol as opposed to supporting intuitive interaction with the end users. Alternatively, interactive and intuitive VQSs should rely on the assumption that the users of the system can and should be able to decide and configure what and how information is displayed in the interface [30]. In our approach, we explore the application of a user-defined domain-specific language (DSL) as the intermediate layer between the end-user and the system’s interface, similar to what has been used in other approaches [31–33]. The term DSL is typically used to refer to a specification language devised to address a particular design problem such as describing a biological process or designing intuitive interfaces. DSLs in biomedical sciences have been used in the past to model and describe biological interactions - Greg, for example [34], was developed to model the various concepts of the Gene Regulatory Mechanism; the development of various formal DSLs has also been driven by the need to orient user’s interaction with bioinformatics applications such as Cell Illustrator [35], GINSIM [36] and Cytoscape [37]. Alternatively, more generic DSLs can be derived from domain-specific biomedical ontologies [33,38] - in the context of semantic VQS design, these DSLs can function as intermediary mechanisms or “glue” between the visual exploration requirements, data available in various SPARQL endpoints and the set of rules and graphic elements necessary to design an attractive VQS. DSLs differ from ontologies in their motivation and size: the term ontology in computer science is used to describe a conceptualization, which may and should entail any number of other ontologies. As such, ontologies can grow indefinitely and a well-construed ontology requires subscription to a set of upper ontologies which adds further non-domain-specific concepts to the ontology (e.g. “Thing”, “continuant”, “occurent”, “dependent continuant”, etc). Alternatively, DSLs are typically much smaller and self-contained as they are designed to address a specific problem and do not attempt to be extensive in their representation. S3QL [31] is an example of a compact DSL devised by domain experts, where the intricacies of specific conceptual domains are captured in the DSL and applied in the generation of a SPARQL query. S3QL has been used to devise VQSs like AGUIA [30]- ontologies are used, in those cases, as sources of controlled vocabularies, but are not applied in the user’s interaction with the VQSs to avoid increasing the complexity of the system and overwhelming the user with unnecessary details. Other VQSs like DistilBio [39], VIQUEN [40] and Cuebee [41], which were devised and tested with
life sciences datasets, assemble the visual model using ontologies. The main disadvantages of these systems are 1) the requirement that users of most of these VQSs need to know the formal ontology schema, which can often be very large and unmanageable [42] and 2) federated querying is not supported. As such, there are advantages to applying a user-designed DSL to configure a visual model, as opposed to an ontology, since the former ensures that some of the cognitive challenges associated with learning a new visual interface are removed by the fact that the system relies on the “user’s own language”.

1.2 Evaluating User Experience with Usability

Our application of a user-designed DSL, as opposed to an ontology, as the intermediate layer is based on two main usability hypotheses - 1) that usage of a DSL makes the interface more intuitive (familiarity) and that 2) the size of the visual model affects the time needed to assemble a query (concept overload). Traditional laboratory studies, in which evaluators are presented with a set of tasks to be completed and the time required to execute any task is measured, along with the number of clicks or errors, exist to evaluate the usability of any system. However, for the effective adoption of a semantic web application in the biomedical domain, it is essential to identify from the beginning, whether the domain users, found the whole user experience (UX) engaging, which is a bit tricky to estimate. Specifically, the term ‘usability’ is used to indicate whether the user can easily, efficiently and effectively configure any system to function according to his requirements, whereas ‘user experience’ is the understanding of how any user feels about using the system [43]. The traditional laboratory studies could be extended easily to collect experiential insights of the evaluator, during the preliminary stages of application prototyping as proposed under [44].

In this report we present our approach for the development of a semantic VQS prototype named ReVeaLD, which we used to collect evaluation data from domain experts. ReVeaLD (Real-time Visual Explorer and Aggregator of Linked Data) relies on a DSL-driven and easily configurable visual exploration method that translates graphically assembled queries into its SPARQL equivalent and returns the corresponding query results. The SPARQL query engine that ReVeaLD relies on is a SPARQL federated framework that transforms a simple SPARQL query into its federated equivalent by relying on a dynamic catalogue of more than 50 life sciences
datasets. In the following sections we will address the requirements for visual exploration in biomedical domains and the federated architecture of ReVeaLD. A key feature enabled by the DSL-driven design is the ability to define, for specific data elements (e.g. chemical elements or biological pathways), the visual display interface that makes more sense to the domain users (e.g. 3D molecular structure or a pathway diagram), as opposed to generic interfaces. We showcase the different usability features of ReVeaLD, whilst addressing four common requirements identified by domain experts in cancer chemoprevention research:

1) Retrieving information for simple queries. For example, toxicity and referred publications of chemopreventive agents, derived from ‘pomegranate’ (source) and are responsible for affecting pathways involved in ‘Estrogen’ production,

2) Assembly of Dataset-specific SPARQL queries. For example, discovering drugs, with molecular weight below 1000, and supplementary information, from DrugBank, which are used to treat diseases reported in Diseasome datasets entitled ‘Colon Cancer’,

3) Enabling user-specific extensions to the underlying DSL. For example, addition of a new concept named ‘RNAMolecule’, a new string property ‘containsNucleotide’ associated to this concept indicating the symbol of the nucleotide (A, C, U or G), and reuse of a concept termed ‘DNAMolecule’ extended by another researcher,

4) Secure querying of datasets generated from private experiments. For example, querying molecules whose functional activity, as qualitatively assessed in the assays described in a study [45], produced desirable results for potential cancer chemoprevention, and correlating them to similar molecules mentioned in public knowledge-bases.

We evaluated ReVeaLD using a game-based evaluation method for measuring the intuitiveness of the tool. The evaluation focused on resolving two usability concerns regarding the DSL-driven design of ReVeaLD: 1) Does familiarity of the users with the DSL affect the time needed to formulate the query, and 2) Does a constrained DSL (smaller DSL), lead to less time needed for query formulation? Finally, we will discuss the applicability of ReVeaLD beyond the cancer chemoprevention domain in which it has been tested and its portability to other semantic web frameworks i.e. although the ReVeaLD proof-of-concept relies on a federated SPARQL engine, it can also support visual query assembly in more traditional SPARQL endpoints.
2 METHODS

ReVeaLD is implemented as an exploration platform and the DSL used as a proof-of-concept was generated by extracting the concepts and properties in CanCO\(^1\), a biomedical semantic model collaboratively devised as part of the EU GRANATUM Project [46]. CanCO was initially devised as a lightweight model and aggregates the set of concepts and properties (Query Elements - QE) relevant to the domain of cancer chemoprevention. CanCO was formalized in RDF/OWL; its design was identified as a part of a collaborative effort involving both computer scientists and biomedical experts. The aim of devising such a model was the identification of a common language which could be used easily by partners in the project to exchange, handle and display relevant concepts across different modules. Since CanCO is the product of a concerted effort by some of the best minds in the domain of cancer chemoprevention, it was expected that the high level concepts and properties identified therein would be more relevant and intuitive for cancer chemoprevention experts than the models identified uniquely by computer scientists. CanCO Universal Resource Identifiers (URIs) were aligned to widely-known biomedical guidelines, standards and controlled vocabularies available through BioPortal [13]. CanCO was used in the integration of various heterogeneous datasets and to anchor other GRANATUM applications to a single model. Another concern while developing CanCO was the creation of a constrained space of queriable concepts and properties. The set of QE for CanCO was identified from four major areas - Literature, Life Sciences, Experimental Data and In-Silico Modeling. In addition to the QE provided by CanCO an extended version of the DSL was used in ReVeaLD which included the 1248 concepts and 1255 properties harvested and linked to the QE from 53 Linked Biomedical Data Sources (LBDS) according to the methods described in [25] (the Life Sciences Linked Open Data - LSLOD catalogue). This entails all the datasets deemed relevant for cancer chemoprevention, including DrugBank, Chebi and Reactome.

\(^1\) http://bioportal.bioontology.org/ontologies/3030
2.1 DSL Visual Representation

Domain experts outlined the necessity of a concise graphical representation of the DSL, depicting the available concepts and associated properties that would provide the rudimentary user a clear overview of the queriable concepts, and guide him towards intuitive query formulation. We reviewed several visual representation methods and came to the conclusion that using a concept map would be the most effective method. A concept map, which is a graphical method representing the relationship between nodes and links [47], has been used in various domains for organizing knowledge [48–50]. It has been proven to successfully enable preliminary learners to easily interpret the query problem, retrieve the knowledge structure represented in their minds for solving the query, intuitively deduce new relations among the concepts embedded in the problem and remember this representation process for future uses [51,52].

![Concept Map](image)

**Figure 1:** A concept map representation of the GRANATUM DSL on ReVeaLD. Concepts are represented as circular nodes, and the literal attributes related to a concept are shown as rectangular nodes. The nodes are connected depending on the relationships between the underlying concepts. Similar concepts are grouped together and represented using the same color. The size of the nodes indicates the number of concepts of LSLOD catalogue, linked to the DSL concept.

Visual models for completing the concept maps were generated from the GRANATUM DSL. Concepts are represented as circular nodes and literal attributes (molecular names, weights, etc.) are represented as rectangular nodes, as shown in Fig. 1. The nodes are connected using labeled arrows based on the relationships of their underlying concepts. Concepts in adjacent ontological layers (e.g. Molecule->Sugar and Molecule->Protein), are grouped together and are
represented using the same color to increase usability. The size of the nodes reflects the number of concepts linked to other knowledge-bases as described in the LSLOD catalogue. A further description of each concept is displayed by hovering over its node. The nodes are modeled as objects in a two-dimensional system using a force-directed layout [53]. This force-directed layout prevents overlapping, as one node constantly repels other nodes based on their size, and facilitates grouping of similar nodes based on hierarchical relationships. As a result, all the concepts are discernible to the end-user.

ReVealLD applies a knowledge-based path discovery method to support the assembly of the query pattern between concepts in the GRANATUM DSL. As an example, ReVealLD visual browser can discover relationships between “assay” and “pathway” (assay--> hasInput--> Molecule--> affectPathway--> pathway), thus emulating browser-based reasoning capabilities, i.e. the GRANATUM DSL extended with LSLOD concepts provides a semantic context that guides the challenging task of building complex and intricate queries. The paradigm empowering this feature is based on applying a shortest-path method (Dijkstra's algorithm [54]) that enables the discovery of links between distant nodes through reusing and mining available links. If intermediate nodes are specified as a necessary output of the SPARQL query (e.g. Molecule), the algorithm will discover the shortest path that includes those intermediate nodes as well.

2.2 DSL Incrementation

Even though the core GRANATUM DSL identifies the set of concepts that are relevant starting points to assemble queries, it is often necessary to extend the list of available query elements to include new terms on a case-by-case basis. As an example, whereas at the time of devising CanCO only a few properties for the concept “Protein” were identified, exploring available data on the LSLOD Catalogue revealed that some properties associated with Protein (e.g. Peptides, Protein-Domain) may be useful for inclusion in the query. Since these additions are relevant for only a few researchers they were not included in the constrained GRANATUM DSL as they would unnecessarily increase the complexity of ReVealLD. For this reason, a DSL Incrementation mechanism was implemented where users can select from the query elements
available in the LSLOD catalogue or create new ones. This mechanism allows the addition of new concepts, relations and literal properties. For each new extension, a URI is generated, along with the user-provided data (name, description, domain, range, etc.). These extensions are serialized using JSON and are submitted using REST [55] POST method to the Incrementation mechanism. The mechanism adds the new extensions to CanCO and annotates it with the appropriate provenance information (i.e. the creator of each extension). When the necessary modifications are made the newly added concepts are included into the visual representation. These manipulations are saved as standalone models which can be later retrieved by authenticated users. The mechanism also supports merging of two user-extended models and creation of a new version containing the changes of both extensions. The Incrementation mechanism is based on the JENA API [56] and is triple-based.

2.3 Graphic Rules

Data is stored in the form of triples (<subject><predicate><object>) in the LBDS. These predicates are referenced using properties mentioned in the DSL. After carefully reviewing these triples, we identified distinct patterns. Based on these patterns, we established a set of rules which dictate the dynamic conversion of the generic textual or numerical content to rich media interfaces, which are visually suitable for biomedical researchers. For example, whenever the string pattern ‘sdf_file’ is present as a predicate, the object of the triple is generally a literal URI, indicating the location of the file which defines the structure of the subject resource (e.g. aspirin). Hence, the 3D structure of the molecule fits the visual model better than the URI. These ‘Graphic Rules’ follow the Event-Condition-Action paradigm [57] and have been defined using the Fresnel Display Vocabulary [58] and the RDF Triggering Language [59]. A Graphic Rule is composed of four major parts:

- A Trigger, usually a regular expression pattern in the predicate or the object (source).
- An Action dictates the replacement of the value contained in the object, with the appropriate blob of content. The action may involve executing HTTP GET requests to external applications or extracting content from web documents and parsing the output.
• The Resource Renderer, a component invoked by the Action, which can easily be plugged in the platform if required, is responsible for the conversion of the generated content.

• The HTML template where the rendered resource is inserted. A second template is also provided, which provides the user with an alternative method to access the object value, if the Resource Renderer fails to convert the content blob correctly.

In the previous example, the Trigger consists of a pattern (‘sdf_file’) and a source (predicate). The Action attempts to read the contents of the URI, contained in the object of the linked triple. If the content adheres to the standard SDF notation [60], a molecular viewer application (Resource Renderer) is invoked to render the 3D structure and is inserted into the relevant HTML Template. The Trigger can also be a combination of various different patterns. For example, when the pattern ‘seeAlso’ occurs in the predicate in conjunction with the patterns ‘http’ and ‘.html’ in the object, the graphic rule assumes that the object is a web document and the Action renders a snippet of this document contained in an <iframe> HTML template. It is relatively easy for any developer to include his own Graphic Rule and faulty content conversions are rendered as simple text.
2.4 Components of ReVeALD

![Wireframe diagrams of ReVeALD’s major User Interface (UI) components]

The Visual Query Builder (A) and the Faceted Data Browser (B). The Visual Query Builder consists of the Visualization Interface, the Query Interface, and an Extension Interface for authenticated users. The Query Interface includes an auto-complete search box, along with a Literal Selector Table, to model the query. DSL extensions made public by other users are shown in an Extensions Table. Faceted Data Browser, coupled with the Lens Dialog, enables intuitive viewing of the results.

While developing ReVeALD, our primary concern was to keep the frontend user interface (UI) as simple and minimalistic as possible, so that the typical clinical researcher is not overwhelmed at the first launch of ReVeALD. To this end ReVeALD essentially consists of two major components, as shown in Fig. 2 - the visual query builder and a data browser that facilitates faceted navigation of data. The visual query builder contains two sections – the visualization interface and the query interface. Both, the constrained DSL as well as the LSLOD Catalogue, are represented on the visualization interface, using the concept map representation described in section 2.1, allowing the user to select the required concepts and assemble the queries. The query interface allows the users to further optimize their query. An extension interface is also embedded in the visual query
builder for authenticated users to extend the current DSL, with newer concepts and properties. Users can authenticate themselves with the GRANATUM Platform using a toolbar control present in ReVeaLD, and load their previously saved queries or publicly available extended DSL models onto the visual query builder. The data browser is also equipped with a Lens Dialog Interface which allows the user to introspectively review each result.

As part of the GRANATUM effort a federated, reasoning-enabled SPARQL engine, the GRANATUM Query Engine (GQE), was implemented and exposed through a SPARQL interface to support the transformation of simple queries like \[\text{(?molecule a granatum:Molecule)}\] into their federated counterpart \[\{\text{(?molecule a chebi:Compound } \cup \text{(?molecule a granatum:Molecule)}\}\] in real-time. The GQE itself makes use of the LSLOD catalogue and semantic links of the type \{ Concept\_A subclassOf QE \}, \{ Concept\_A void:uriRegexPattern stringPattern \} and \{ sparqlEndpoint void:class Concept\_A \}. These semantic links are automatically generated according to the ‘a posteriori integration’ methods described in [25], and are validated by domain experts. The mappings allow identification of all instances of a particular QE (Molecule), and internal selection of only those SPARQL endpoints which contain some data pertaining to that QE. Due to the unpredictable status of public endpoints, the GQE recursively monitors the latency of all the endpoints and uses this information to determine ‘smartly’ which of the previously selected endpoints would be available for query execution. The query processing time at each endpoint is also monitored, and those endpoints which take longer than two minutes to push back the first result have their connections automatically terminated for advanced query optimization and better allocation of the GQE resources.

ReVeaLD relies on the GQE to provide real-time SPARQL federation functionality but can just as easily be configured with any other distributed query processing framework. ReVeaLD uses a REST [55] POST method to submit user-assembled SPARQL queries to the GQE’s read/write interface and query results are retrieved in JSON (JavaScript Object Notation) format and processed in the faceted data browser. Authenticated users can also query their private experimental datasets annotated with the DSL via ReVeaLD. The overall architecture on which ReVeaLD relies – including GQE and the LSLOD catalogue - is represented in Fig. 3.
Figure 3: System Architecture Diagram of the Semantic Cancer Chemopreventive Drug Discovery Platform developed under the GRANATUM Project. ReVealD is a browser-based client application, allowing users to formulate queries intuitively using a visual model of the GRANATUM DSL. These queries are translated to SPARQL and are submitted to GQE, which provides real-time federation functionality. The results are subjected to a set of Graphic Rules, which dynamically assemble media-rich user interfaces, hence presenting results in a format relevant to biomedical researchers (Pathway Maps, Molecular Structures, etc.).

2.5 Technologies

ReVealD is a browser-based client application built using a variety of web technologies including HTML5, SVG, JavaScript, AJAX and JSON. This has several advantages over using application frameworks such as Adobe Flash or Microsoft Silverlight since no further software needs to be installed on the client, thus enabling integration with portable devices. The visual query builder is based solely on SVG (Scalable Vector Graphics) with extensive JavaScript usage. The communication between the client and the GQE relies on AJAX. HTML5 local storage is used to store essential named key/value tokens in the client web browser. JSON Format is used for
rendering the DSL on the visualization interface, as well as for serializing the Graphic Rules, due to its lightweight interpretation in browsers. ReVealLD makes use of various open-source JavaScript libraries: D3.js library [61] is used for data-based manipulation of the HTML DOM in building the visualization interface; ReclineJS [62] is used to provide the faceted browser, enabling a more introspective view of the results; MustacheJS [63] provides the logic-less template syntax used for the extended visualization of the instances. In the current implementation of ReVealLD, the GLMol Molecular Viewer (a WebGL-based Javascript Library to visualize 3D structures of molecules) [64] and the Flot plotting library [65] (to create area graphs of numerical data) are also integrated for domain-specific visualizations.

2.6 Evaluation

We conducted a preliminary evaluation of the GRANATUM ReVealLD platform using a laboratory study based on the ‘Tracking Real-time User Experience (TRUE)’ methodology [66], which has been widely adopted lately by the HCI community for evaluating user behavior in computer games. A competition was conducted in which the biomedical researchers formulated five different real-life queries (Table 1). The number and sequence of steps, the time taken on each step to assemble the final query and the state of the visual query builder (to handle the context-dependent UX evaluation) were recorded. The queries were selected to address most of the features of ReVealLD while at the same time being realistic with use cases provided by the domain users. The tasks were assigned with increasing complexity like levels in a game. All five queries retrieve data in real-time. The evaluation tasks included a combination of single-concept and multiple-concept selections, as well as selecting and filtering a varied number of literal properties. Moreover, the tasks also included scenarios based on query formulation against specific datasets. A compressed version of the LSLOD Catalogue, which contained concepts and properties of only those datasets relevant in the tasks, was used for this purpose to ensure that the first-time evaluators were not overwhelmed.

Table 1: Summary of the Tasks presented to the Evaluators

<table>
<thead>
<tr>
<th>Description</th>
<th>Concepts</th>
<th>Literals</th>
<th>Visual Model</th>
</tr>
</thead>
</table>

| Task 1 | Assays which input ChemopreventiveAgent titled ‘Aspirin’ | 2/9 | DSL (flexible) |
| Task 2 | Chemopreventive Agents, derived from ‘Pomegranate’ Source, which affect Pathways titled ‘Estrogen’, and all the Toxicity details about these agents. | 4/5 | DSL (flexible) |
| Task 3 | All the details about Uniprot Journal Citations titled ‘Mouse’ | 1/7 | LSLOD Catalogue |
| Task 4 | IUPAC Names, Inchi Keys & SMILES notations of Chebi Compounds with Mass less than 200 and have a ring-like structure (have the word ‘cyclo’ in the title) | 1/5 | LSLOD Catalogue |
| Task 5 | Diseasome Diseases labeled Colon Cancer which have possible DrugBank Drugs with Mol. Weight less than 400 | 2/50 | LSLOD Catalogue |

The entire evaluation process was conducted as a game to encourage users to engage with ReVeaLD and also allow us to gain a better understanding of user behavior in the ReVeaLD interface. Each task was scored by the number of correct query elements selected and weighted negatively whenever hints were used during the process. Points were deducted from the score when the user added a query element that was not required, or when constraints on the literal properties were not set. After all the tasks were completed users were given a score as the ‘Number of Granatum Seeds’ (final score) and were notified of the time taken after each task. The paths (sequential selection of concepts and literals) followed by each user to formulate the queries were logged using Google Analytics [67], along with the time taken on each consecutive step. The accuracy of each evaluation was calculated using the correct sequential selection of the required concepts and literals. We also recorded whether or not the user is able to assemble the correct query on the first try. Additional information about our users was also logged, such as the browser/OS configuration and screen resolution, and the server response times of the platform in order to assess whether hardware bottlenecks could be responsible for difficulties faced while assembling the query. All users were informed of the information collected and agreed to participate in the evaluation process.
For evaluating our two main hypothesis indicated under Section 1.2, ‘steps’ taken and the time spent on each step per user were aggregated into two categories: total time spent on the Visualization Interface (concept selection), and total time spent on the Query Interface (literal selection). This was based on classifying each step as belonging to one or the other category. Outliers were removed – these were caused by events such as a server/network delays. At any point, the Visual Query Builder would contain a limited set of query elements (concepts or literals), and the time spent in choosing a single one was measured. Best-fit regression models (linear, logistic, exponential or polynomial) were calculated and missing values were ascertained. Whereas for concepts, regression was calculated from time taken to select a single concept from a pool of X concepts, there was a minor modification in Literal selection - as more advanced users were accustomed to selecting all the required literals in a single step, we used the percentage (%) of literal properties (No. of literals selected/No. of literals available) selected for calculating regression. The relative familiarity of the evaluators towards the usage of query elements from a constrained DSL against those extracted from the LSLOD Catalogue was hence studied.

To gain insight into whether our evaluators found the experience engaging we were able to record the user retention (if the same user participated in any task more than once), the number of attempts made to assemble any query and the step after which the user exited the competition, using Google Analytics. To augment this behavioral data we presented a brief questionnaire to the evaluators on a later stage to assess the UX and usability of ReVeaLD as an integrated component of the GRANATUM Platform. The attitudinal data so collected, also shed direct light on issues like the usefulness of the retrieved results to the user (usage in virtual screening platforms), and their expectations. The questions given to the users have been listed under Appendix II.

2.7 Availability

The ReVeaLD platform has been integrated as a specific component of the EU GRANATUM Project, and can be accessed at http://srvgal78.deri.ie:8080/explorer. The documentation, application walkthrough, as well as some sample queries, has also been provided with this platform under the ‘Help’ section. A video screencast, showcasing three usage scenarios of the
ReVealD platform, in context with cancer chemoprevention data discovery and extension, can be viewed at http://www.youtube.com/watch?v=nZqjQekKGGY&hd=1. The ReVealD platform was developed using JavaScript and SVG and was deployed as an independent frontend to the GQE but it can be redirected to any other SPARQL endpoint. It communicates with the GQE using SPARQL 1.1 protocol as well as with the set of GRANATUM REST APIs, documented at http://srvgal78.deri.ie:8080/wsdocs/. The default SPARQL interface of the GQE is accessible at http://srvgal78.deri.ie:8080 and the GRANATUM platform can be accessed at http://granatum.org/bscw/. Testing and evaluation of ReVealD was performed by measuring the usage and the response times as well as the visual model assembled by users to formulate his query, using the Google Analytics Core Reporting API.

3 RESULTS

ReVealD is a prototype for a linked data visualization and exploitation designed to lower the entry barrier for Linked Data adoption in life sciences domains. To achieve this, ReVealD applies a novel user interaction paradigm enabling three core requirements: i) assembly of federated SPARQL queries without the requirement of specifying the data source; ii) configuration of a visual model based on a user-defined DSL; iii) intuitive data navigation by combining semantic web technologies with principles from the area of HCI. We will describe each of these three requirements and how they were implemented in ReVealD. The need for addressing requirement i) was thoroughly discussed in [25,68] and is related the exponential growth in the size of data and the number of sources in life sciences domains. The need for requirement ii) is derived from the observation that researchers are more willing to learn and use a new tool when they are familiar with the language used to convey instructions or, in the case of ReVealD, to convey the visual model. In this section, we will describe the ReVealD prototype and the method used to evaluate requirements ii) and iii) based on gamification principles.
3.1 Description of ReVeaLD

In ReVeaLD, the user is guided in the formulation of complex and federated SPARQL queries using a touch-and-click "smart" interface that is fueled by a visual graphical model. Our primary concern was to keep the frontend user interface as simple and minimalistic as possible to avoid overwhelming the typical user (e.g. clinical researcher and cancer biologist). The visual model itself is derived from the user-configured DSL which can be configured and incremented by the domain experts. The DSL can be derived from any ontology; however the number of QE in the DSL is constrained to avoid cluttering the interface. QE in the extended DSL are linked (via the "subClassOf" predicate) to concepts extracted from the schemas used in over 50 linked life sciences datasets. Also to reduce cluttering, property labels are hidden by default but can be made visible through an option on user preferences. The default visual model displays only the nodes from the DSL but the advanced mode supports a visual model that includes all concepts and properties extracted from the LSLOD catalogue. This means that default queries target only QE identified in the DSL (e.g. granatum:Molecule) but more advanced queries can be made by selecting concepts described in specific SPARQL endpoints (e.g. chebi:compound). The visual models are loaded in the main visualization interface (Fig. 4 (A)) and the user can assemble his query visually by clicking on the nodes (Fig. 4 (B)). We describe this further in the section 3.2.
Figure 4: The visual model of the DSL (A) is rendered on the Visualization Interface (C) of ReVeaLD, which can be used by the user to assemble his query model (B). Alternatively, the user can assemble the same model, using the auto-complete box (G) in the Query Interface (D). Concepts can be selected directly from the LSLOD Catalogue by deselecting the ‘Flexible’ option (H). The Constraint Selector Table (I) allows him to set textual and numerical filters on the literal properties of selected concepts and he can choose the maximum number of results desired, and the format (J, K). A toolbar is available (L), which incorporates extensive Help (E) on usage of ReVeaLD, and quick access to saved query models. An Extension Interface is also provided for authenticated users to increment the visual DSL.

Events in the main visualization are linked to events in the query interface (Fig. 4 (D)): to initiate query assembly, users can either pick a node from the visual model or alternatively, pick a QE from an auto-complete box in the query interface (Fig. 4 (G)), where a few letters will prompt a listing of concepts matching those letters as a dropdown menu - filtering using this option will cause the visualization interface to be refreshed with the selected concept in the center. The options in the auto-complete box are then dynamically refreshed to display direct relationships to selected concepts. Concepts represented on the visualization interface always include suggestions.
about similar concepts and possible query properties. Properties of concepts present on the visualization interface where the values are literals (rectangular nodes in Fig. 4 (B)) are listed on the query interface in the constraint selector table (Fig. 4 (I)), where the user can select and set filters (molecularWeight ‘Less Than’ 200). Only properties selected in this table are returned in the result set but a ‘Select All’ option is available for this table. The number of maximum results can be selected and users are given the choice of downloading or examining the results in the faceted data browser (Fig. 4 (J, K)). The result of this process is a visual representation of a SPARQL query - a visual query model. Notations used for variables in this model were inspired from vSPARQL [20] with slight modifications: i) the inclusion of possibly related or similar concepts in the query model, as suggestions, ii) prefixed node labels of variables, e.g. ‘?x0_ChemopreventiveAgent’ - the addition of ?x0 is due to the need for unique variable names (auto-incrementing numbers) in the SPARQL formulation, iii) highlighting of variables which had explicit filters set using a ‘Red’ color, and iv) exemption of unwanted notations specifying graph constraints or union/optional graph patterns. This visual query model is internally translated to a SPARQL SELECT statement as shown in Table 2. A ‘Save Query’ Button allows logging these queries associated with the ReVeaLD user account, and can be accessed later using the toolbar (Fig. 4 (L)). Authenticated users can also make changes in the DSL via an ‘Extension Interface’ (Fig. 4 (F)) - the user can increment the DSL by adding nodes or links to the visual model. The extended DSL can be stored under the user’s account and publicly available DSL by other users can be accessed and extended via a non-intrusive pop-up dialog or merged into the default model. Usage Scenario 3 describes this scenario.

Table 2: The translation of a Visual Query Model to its SPARQL SELECT Statement by ReVeaLD
SELECT DISTINCT * WHERE {
  ?x0_Journal_Citation a uniprot:Journal_Citation ;
  uniprot:pages ?x1_pages ;
  dc:identifier ?x2_identifier ;
  uniprot:volume ?x3_volume ;
  uniprot:date ?x4_date ;
  uniprot:author ?x5_author ;
  rdf:void ?x6_void ;
  granatum:title ?x7_title .
  FILTER( regex( xsd:string( ?x7_title ), "mouse", "is" ) ) .
}

Figure 5: The Data Browser integrated with ReVeaLD shows the results obtained from query execution in a tabular (grid) format (A). The user also has a choice to create a line plot of numerical data by switching to Graph format (B). The interface allows 'search' and 'filter' of these results (C and D) and facets on the dimensions are automatically computed (E) and available for slicing. Clicking any resource (highlighted in Blue) in the results dataset, triggers the launch of a Semantic Lens Dialog (F), which shows the knowledge...
graph of the clicked resource using rich domain-specific interfaces (H). This graph can also be downloaded in the desired format (G).

Queries are sent to the GQE and results are rendered in a table (Fig. 4(A)) by default – columns can be sorted, hidden and arranged. Results in each column can be further filtered by defining a ‘Text’ or a ‘Range’ for string or numeric values respectively. The results can also be searched and “slices” of data can be created based on preferred terms. ReVeaLD automatically computes facets from the results obtained and allows the user to further filter the data – the data browser is configured for the DSL fueling the visual model where specific concepts and predicates (e.g. Molecule) trigger the assembly of domain-specific interfaces (e.g. 3D structure visualizer of a molecule). The results retrieved as a direct consequence of the query execution are subjected to a set of Graphic Rules which dictates the dynamic assembly of the Lens Dialog whenever any resource is clicked. Results can either be resources (referenced by URIs) or literals. The faceted browser responds differently to each type. A listener pattern was created for resources that prompt the launch of a non-intrusive ‘Semantic Lens Dialog’ whenever any resource is clicked. The SPARQL query (SELECT * WHERE {<clickedURI> ?p ?o}) is executed to retrieve all triples (<clickedURI>-predicate-object) where the selected resource has been explicitly used. The assembly of the media-rich user interface of the Lens Dialog is handled by the set of graphic rules which were defined for a subset of resource types. The Lens Dialog enables multiple resources to be displayed simultaneously and compared using tabs. This information can also be downloaded.

3.2 Usage Scenarios

The requirements described under Section 1 were identified and refined by the biomedical domain experts involved in the GRANATUM project and the corresponding queries (shown in Appendix I) were emulated in ReVeaLD using visual query models. The following usage scenarios provide a detailed step-by-step approach to address each of these requirements.

3.2.1 Scenario I

Fig. 6 (A-I) shows the set of steps undertaken to assemble the SPARQL Query required to address the Requirement I in ReVeaLD.
Figure 6: The sequence of steps required to formulate a query, using ReVeaLD to determine the toxicity information about chemopreventive agents derived from 'Pomegranate' and which affect Estrogen-related pathways along with their references in published work. To select the desired concepts, the user can use the auto-complete input provided in the Query Interface (A, B), or click them on the Visualization Interface (C, E, and G). He can set the required filters or constraints and can also select the literal attributes he requires in the result set, using the Constraint Table present in the Query Interface (D, F and H). Finally, he has the option to set the maximum number of results required and the format of the results (I).

- The researcher could assemble the SPARQL query by typing ‘Chemopreventive Agent’ in the auto-complete search box provided in the query interface. As soon as he begins typing, a list of concepts containing the letters entered is shown (Step A).

- After selecting ‘Chemopreventive Agent’ all the relationships and the literal properties of the selected concept are presented to the user for further selection (Step B). In this case, he is attempting to find all possible relationships between a ‘Chemopreventive Agent’ and a ‘Pathway’. Since he may not know how many links separate the two or what their names are, ReVeaLD allows discovery of the shortest link between the two by typing their names in sequence in the auto-complete box (Step B and C).

- A filter on the pathway title can then be set to select only those titled ‘estrogen’ (Step D)
• To filter the source, the concept ‘Source’ connected to ‘Chemopreventive Agent’ can be selected and a filter can be set in the Query Interface, with the string ‘pomegranate’, on its common name (Step E and F).

• Finally, since the goal of the query is to retrieve toxicity information and related publications, he selects the ‘Toxicity’ and ‘PublishedWork’ nodes and can chose to return any specific property about those (Step G and H). He retrieves the data by clicking “Get Results” (step I).

   It is worth noting that each step involved in assembling a query adds extra numerical parameters to the query URI, which reflect references to concepts and relationships in the DSL - the final query becomes an effective URI which can be shared with collaborators by simply sharing the URI. Its invocation will trigger the automatic assembly of the visual query model. As an example, the visual query model in this use case can be assembled by pointing the browser to:


   Triple patterns in the SPARQL query are represented as numerical references in this compact URI notation - for example, the triple pattern (?x0_ChemopreventiveAgent granatum:affectPathway ?x1_Pathway) is represented as 30.25. The ‘filters’ parameter specifies a chained representation of each FILTER statement- for example, 25.91.c.estrogen translates to FILTER( regex(xsd:string(?x5_title ), "estrogen", "is" )). A dataset-specific query would be represented in a similar way with the distinction that the numerical references would refer to the extended LSLOD catalogue representation. Even though the URI itself is not presented in a human-readable format, it can be argued that this native format serves as an excellent method for collaboration and query exchange.

3.2.2 Scenario II

One of the most predominant applications of ReVeaLD is the assembly of SPARQL queries traversing more than one dataset (Requirement II). Without ReVeaLD, the complexity of this problem (in our example, discovering the drugs matching the exact specifications) would require the researcher to be thoroughly familiar with both SPARQL federation methods and the schemas of the relevant datasets (Diseasome and DrugBank). Most of the steps are similar to the ones
described in Fig. 6 and thus will not be detailed here. The unique distinction is the deselecting of the ‘Flexible’ option in the Query Interface (Fig. 4 (H)), which enables the specification of DrugBank/Diseasome-specific data.

### 3.2.3 Scenario III

One of the core requirements of ReVeaLD was the need to increment the DSL with novel concepts such that they could be queried and used to annotate data. To address the example mentioned under Requirement III using traditional methods, a SPARQL UPDATE Construct is required to increment the DSL with the new concepts/literals (the UPDATE contains the addition of the ‘DNAMolecule’ concept because it is not feasible to manually merge the two models).

**Figure 7:** To increment the DSL using ReVeaLD, the researcher has to authenticate himself with the GRANATUM Platform (A). He can add the required concepts and relationships using the Extension Interface, shown after authentication (B, C and D). The user is notified after a successful Incrementation of the DSL, and the new concepts are rendered in real time (E and F). He can also merge his model of the DSL, to other publicly available versions, to introduce concepts previously extended by other researchers (G and H).
With GRANATUM ReVeaLD an easy click-based mechanism can be used to assemble that same query. Moreover, it is possible to use a colleague’s extension with detailed information about changes she may have made when compared with the common model. The sequence of steps in Fig. 7 (A-I) illustrates this process.

- After authentication of the user with the GRANATUM platform (Step A), the Extension Interface of ReVeaLD is opened. The concept ‘RNAMolecule’ can be added to the core ontology from which the DSL is extracted - a description as well as a Parent class (superClass) can be identified by opening a drop-down list with the existing concepts in the ontology (Step B).

- The extension interface can also be used to add a name and description for a new relationship/property as well as to select the rdfs:domain and rdfs:range (including string, integer, date-time or Boolean for literal properties). In the example, the rdfs:domain is the newly added concept ‘RNAMolecule’ and the rdfs:range is ‘xsd:String’ (Step C). When the DSL is incremented (Step D) changes are available in the visualization interface (Step E and F). This model is also saved into the user’s account for future queries or annotation of experimental datasets through GRANATUM-associated tools.

- To include changes made by another researcher, the appropriate extension is selected in from a list of publicly available models, shown in a non-intrusive dialog (Step G). A new model is created, which includes the concepts extended by both the researchers (Step H).

### 3.2.4 Scenario IV

One of the primary uses of ReVeaLD is in the secure querying of datasets generated from cancer chemoprevention experiments by various researchers. These datasets are semantically annotated using the CanCO Semantic Model, and are available as password-protected SPARQL endpoints since users must often authenticate to access them. In the example cited under Requirement IV, the researcher knows that the information present in the study [45] has already been RDFized, and he has the necessary permission required to query this dataset. The SPARQL query necessary to address this use case scenario is highly complex in nature, and would only provide a URI reference to the compound similar in Chebi, if executed against the conventional SPARQL engine. However, using ReVeaLD, the researcher can formulate the final query model (Fig. 8 (A)), after
some minor DSL Incrementation (steps described in Fig. 7). The steps of query formulation are similar to those shown under Fig. 6 and will not be discussed here. The researcher can just click on the Chebi URI reference mentioned in the result set obtained (Fig. 8 (B)), and additional information is available in the Lens Dialog (Fig. 8 (C)). Using the highly configurable interface of ReVeaLD, several of these molecules can be investigated simultaneously.

Figure 8: To determine the assays mentioned in a study, detailed in a publication and to find out the molecules, whose functional activity for cancer chemoprevention was qualitatively assessed in those assays, one requires to formulate a query model as shown in (A). After query execution the results as shown in (B) are obtained which also provide references to similar molecules in Chebi and other knowledge-bases. Additional information is just a click away and can be visualized in the embedded Lens Dialog (C).

3.3 Evaluation Results

The evaluation focused on addressing two main usability concerns highlighted under section 1.2. Using the methodology described in the section 2.6 we measured time taken for formulating queries attached to the delineated tasks (1-5). The total number of unique evaluators, which included both the biomedical researchers and the computer scientists, who completed at least one task successfully, was 40. The distribution across different tasks was 40, 36, 28, 28 and 25 respectively (user exit at different tasks). The number of evaluators, who made multiple attempts at assembling the query model faster (user retention), was 24, 24, 18, 16 and 16 respectively. 24
evaluators responded to the questionnaire which was presented later, the results for which are included in Appendix II.

We measured task complexity (Table 1) in terms of concepts/literals to be selected from a larger universe of query elements available. We found that our users had an accuracy of approximately 80% in selecting all the required concepts using ReVeaLD, and 65% when selecting the required literals. As shown in Fig. 9, all the tasks were concluded by our evaluators in less than 2 minutes (120 seconds) on an average. The traditional approach to SPARQL query formulation would easily take even an expert Linked Data Technologist more than 2 minutes due to the need to retrieve data from 53 linked data endpoints and several schemas. The total time includes the initialization time of the interface (blue), the selection of concepts (red), and time spent on selecting literal properties (green). The time taken to initialize ReVeaLD by loading the LSLOD Catalogue (Tasks 3-5), is higher when compared to loading the DSL. This time decreases when the LSLOD Catalogue is cached in the browser’s local memory (after at least 1 task has been performed). The slight increase in loading time for Task 5 can be attributed to the fact that some users tried to evaluate Task 5 first before any other tasks, as it was shown as a walkthrough in the user’s guide. At first glance these results also suggest extra time to select concepts and literals from the LSLOD Catalogue when compared to the DSL. The exception in Task 3 is caused by the ability to “Select All” literals with a single checkbox, which is faster than individually selecting multiple literals.

![Chart showing average time taken for different tasks]

- Task 1: 2 concepts, (2/9) Theory DNS, DNS
- Task 2: 4 concepts, (5/12) Theory DNS, DNS, LUSO Catalogue
- Task 3: 1 concept, (7/7) Theory LUSO Catalogue
- Task 4: 1 concept, (5/20) Theory LUSO Catalogue
- Task 5: 3 concepts, (3/20) Theory LUSO Catalogue, LSLOD Catalogue (Walkthrough)
**Figure 9**: The average evaluation times registered, for the successful completion of any task. This time could be divided into the time taken to load the visual representations (blue), the time taken to select the concepts on the Visualization Interface (red), and the time taken to select the literals (green).

The scope of this evaluation was in testing the usability of the visual query builder – as such, time spent in each interface was correlated with the complexity of queries measured by the number of concept/literals to be selected from a larger population. From the approach described in section 2.6, two exponential regression models were obtained (Fig 10A): average time for concept selection from a population of X concepts in the Visualization Interface from the DSL concepts ($R^2=0.755$) or from the broader LSLOD catalogue ($R^2=0.873$). It should be noticed that the LSLOD Catalogue is composed of over 1200 concepts whereas the constrained DSL contains only about 80 concepts. Two polynomial models were derived to explain the average time required to select a percentage of literal properties via the Literal Selector Table (Fig. 10(B)): when using either the DSL ($R^2=0.862$) or the LSLOD Catalogue ($R^2=0.896$). The chart in Fig. 10(A) is truncated to 150 concepts for improved visibility.

**Figure 10**: Extrapolated times taken to select a single concept from a list of concepts available (A) or a given percentage of literal properties (B), were obtained using regression models, for both the constrained DSL and the LSLOD Catalogue and were charted for analysis. These plots prove the role of increased familiarity of the evaluators, towards their proposed DSL, in the process of query formulation.

From Fig. 10A, it can be seen that the number of concepts available for selection clearly affects the speed at which a specific concept is discovered - particularly when the user is not familiar with the schema (Red line). Although this difference per user is only a fraction of a minute (between 10-15 seconds at any point), as the complexity of the query increases the extra time...
becomes significant in the overall result. These results point in a clear direction: that the increased
time to select query elements from the LSLOD Catalogue is not only due to its size, but also due to
the user’s unfamiliarity with it. When the number of concepts available for selection is the same,
the time to select one concept from the LSLOD catalogue is still higher. The time to select a literal
from a pool of literals also varies between DSL and LSLOD Catalogue (where both familiarity and
size are different). The polynomial regression in this case considers the sudden drop in time for
selection of all literals (100%) as an outlier. This is caused by the availability of the ‘Select All’
checkbox, which makes it faster to pick all literals than to individually select each one. These
results, paired with the observation (not shown) that more accurate evaluations were obtained for
the first 2 tasks (DSL), validates the assumption that the evaluator’s familiarity with their proposed
query elements plays a major role in the query formulation process and increases the intuitiveness
of the platform.

Summarizing attitudinal responses from the questionnaire, combined with the behavioral
statistics (user retention and exit) and analyzing the sequential steps taken by each user
(observations not shown) to formulate a visual query model for any outlined task, provided us with
essential insights regarding the UX of ReVeaLD. It could be claimed that the users of the platform
found the idea of query formulation using visual cues (click-input-select) pretty intriguing, with
some users attempting the same tasks multiple times in competitive spirits. As seen from the
behavioral statistics, the users preferred query formulation using the DSL more than the LSLOD
Catalogue. Also, whereas the users found the selection of the relevant DSL concepts particularly
easy, the refining of the visual query model, using the Literal Selector Table was somewhat found
less intuitive. In particular, the sequential steps indicated that either the user forgot to select the
property altogether, or set the incorrect filter (string versus numeric). The survey responses
showed that while ReVeaLD was found to be very useful and convenient for biomedical knowledge
discovery, there was still scope for improvement in terms of the quality and quantity of the retrieved
results and the ease of query formulation.
4 DISCUSSION

4.1 SPARQL and Biomedical Research

Even though ontologies and RDF have been used to structure and represent life sciences datasets on the web over an extended period, SPARQL is a relatively new technology for biomedical researchers and requires a steep learning curve. This could act as a major barrier, preventing the users to readily access the linked data sets. As a result, agreeing on a standard solution involving the usage of a DSL of constrained concepts simplifies the process of assembly of a generic SPARQL query, not only due to familiarity but also because it enables a less cluttered interface, as illustrated in the results section (Fig 10). Using DSL designed and specified by the domain experts as the source of the visual model ensures that the set of visual cues displayed is constrained by the rules defined by the user, thus circumventing the cognitive challenges imposed by the complexities of learning a new language/interface. The users can use the method presented to create, for example, a model specifying the concepts they wish to access through the query interface. The set of query elements defined can either be selected from a list of concepts extracted from various heterogeneous data schemas (e.g. our LSLOD catalogue) or more abstract terms created as part of a formal model, which are linked to concepts in a catalogue. This method reduces the visual cues to a manageable number. Biomedical, bioinformatics and other domain-specific applications can rely on these DSL for linked data operations since the DSL syntax is internally translated to SPARQL through a set of logic rules. Our preliminary evaluations also appear to support the assumption that the familiarity of the domain users with the DSL facilitates query formulation by accelerating query assembly and increasing its accuracy.

4.2 Increasing intuitiveness by blending HCI with Semantic Web

The involvement of prospective end-users in the early stages of the application prototyping, specifically the constructive collaboration of the biomedical application developers with the domain researchers, has facilitated the adoption and increased intuitiveness of our semantic search platform in the cancer chemoprevention domain. The DSL used as a proof-of-concept in ReVealD
was derived from the CanCO semantic model [46], wherein domain experts identified key query elements relevant to their research. The computer scientists aligned them to public data models and the resultant mappings were again validated by the domain experts [25]. The LBDS were decided by the experts depending on their importance in this field. The conceptualization and the development of ReVeaLD were primarily governed by user requirements at various stages, resulting in the implementation of several UI features ranging from the DSL visual representation, to the domain-specific visualization of instances. The usability and UX insights retrieved using an evaluation method widely adopted in the HCI community, will steer future development.

A long term goal in the development of ReVeaLD is evolving into a system which minimizes the barrier between the cognitive model of what a biomedical researcher wishes to know and the tactical understanding of the semantic query platforms, in particular concerning the assembly and execution of SPARQL queries. Even though the concept of VQSs is not new, as discussed in the section 1.1, most rely on large and formal ontologies for query assembly. None of the existing systems have a provision for real-time federated querying with domain-specific representations. As opposed to other Linked Data visualizers, which enable visualizing and browsing a limited set of data instances, ReVeaLD lowers the entry barrier significantly for cancer chemoprevention scientists by supporting visual interaction with the DSL towards the assembly of SPARQL queries. This is particularly relevant in Life Sciences where the main motivator behind the usage of LD is the ability to assemble queries that traverse several datasets containing structured data. ReVeaLD does not assume \textit{a priori} knowledge either in the field of semantic web technologies (which fuel its data aggregation capabilities) or in the structure of the data. ReVeaLD combines the salient features of both faceted navigation of multi-dimensional data, employed by Exhibit [28] and Lens-based viewing of individual data instances, employed by data browsers like LENA [69] and SemLens [70]. While developing ReVeaLD, we have taken utmost precautions to overcome challenges in using these systems such as the need for the high-level configuration of datasets and technical expertise of the user. The dynamic assembly of the Results Data Browser is governed primarily by the DSL with no user intervention required.
4.3 Towards a Generic Knowledge Discovery Platform

It is worth noting that our current implementation of ReVeaLD follows the soft-coding practice of software development. The collaborative methodology [46] used to create CanCO can also be used to create other DSLs in different domains. The DSL presented to the researcher while formulating his query is not directly embedded into the application. Instead the client browser interprets the DSL at runtime, as an external configuration file. As such, it can be easily replaced by a different DSL, and basic syntactic validation is carried by ReVeaLD. This functionality could be extended to a user-friendlier format in future, where the user can directly select the DSL of his choice by clicking on the options available. The LSLOD Catalogue is loosely linked to the ReVeaLD platform and is not, by itself an essential component, for the query formulation process. We have included it to provide our users with an alternative method to query dataset-specific concepts and to aid in incrementation of the DSL. The semantic rule templates which govern the query transformation to its federated alternatives in the GQE have been automatically generated using the ‘*a posteriori integration*’ methodology mentioned in [25]. It is relatively easy, to link any new data source, as required, without affecting the overall architecture. The scalability of the GQE has been tested against 105 SPARQL endpoints (query processing results are not shown) at a given time irrespective of their latency or uptime. As the GQE has not been intricately integrated with ReVeaLD, it can easily be replaced with another federated, reasoning enabled SPARQL engine which has shown superior results.

Moreover, the Graphic Rules, which propel the automatic assembly of graphics in the Lens Dialog, are defined using standard RDF Schema and used as input to the ReVeaLD platform as an external file. There are two major benefits of this approach: 1) The abstraction of various interface settings, which you can just append to this document, rather than embedding directly into the source code. This enables any authenticated user with a fair knowledge of RDF to include new rules, and plug-in better visualization libraries; 2) The consideration of manual coding errors, while compiling the Graphic Rules. The ReVeaLD approach ensures that the interfaces are displayed, even in event of errors and the media rendered by the damaged Graphic Rule is replaced by the textual value of the converted triple. As we have shown in the usage scenarios, whenever the
underlying model of the DSL is modified, the changes are instantly reflected in the Visualization Interface and these new concepts are available for querying. Due to these modular features it becomes easy to extend the ReVeaLD application to be the knowledge discovery platform of choice in various other domains.

4.4 Limitations and Future Work

One severe limitation of the ReVeaLD Platform is the extreme reliance of the query formulation process on the query elements. To retrieve any information from the LBDS the biologists have to mandatorily select a primary concept after which he can set subsequent filters on the literal properties associated with the selected concept. This does not allow the biologist to make generic queries against the LBDS. As a result generic searches like ‘aspirin’ or ‘pomegranate’ are not supported and he has to make a preliminary selection of the ‘Drug’ or ‘Source’ concept respectively and then set a filter on its title. This limitation is being addressed in the next version of ReVeaLD by building a corpus of title references of different biological entities and using it as a training set for named-entity recognition. There is currently no support for NL-based queries in ReVeaLD, preventing users from formulating search requests based on labels of entities such as is allowed by GINSENG [71]. ReVeaLD depends on the LSLOD Catalogue which links the concepts in the biomedical domain to the core GRANATUM QE. Even though the LSLOD Catalogue has been successful in linking 53 publicly available datasets, we cannot assume that this list is exhaustive.

The query execution component is also not integrated with ReVeaLD and the formulated queries are executed against a predetermined set of endpoints. As a result whenever there are issues with the latency or the functionality of the endpoint, the user cannot get any results from that respective dataset. Due to the lack of instance-level alignment in the GQE, i.e. identifying the same entity referenced using different URIs in different datasets; retrieved query results sometimes may contain duplicates. Addressing this is beyond the scope of this report. We plan to use the approach proposed under [73] to produce schema-level mappings to assist instance-level coreference resolution at a later stage. We plan to implement and integrate a recommendation system with ReVeaLD which would show a list of suggestions of similar instances based on some generic properties (Inchi keys, etc.) that remain consistent across datasets. The system would be based on
the application of Fuzzy SPARQL Queries [72]. Finally, even though currently available VQSs have their limitations as mentioned under section 1, we wish to carry out a comparative evaluation, with these systems, by configuring them with sample datasets and our domain model, conforming to the above use cases. We hope to leverage the usability of ReVeaLD by incorporating the salient features provided by them.

5 CONCLUSION

The Life Sciences Linked Open Data (LSLOD) Cloud is a culmination of complex and structured datasets accumulating biomedical knowledge over years. This paper describes an HCI-based approach towards facilitating biomedical researchers to intuitively formulate queries, targeting multiple datasets simultaneously, to discover relevant knowledge from this LSLOD Cloud. We developed a web-based application to meet the urgent need of the researchers to flexibly mine the Linked Biomedical Data Sources (LBDS), as well as their private datasets, using terminologies specific to their domain, in real-time, and ‘make sense’ of the information retrieved. After carefully reviewing the properties of various resources in the LBDS, we established a set of graphic rules for the autonomous assembly of contextually aware media-rich user interfaces. We also integrated a mechanism to increment this domain-specific language (DSL), which primarily drives our application, so that the authenticated users can extend or merge new concepts and relationships, as per his needs. The user would also be able to use these incremented terms to query or annotate their experimental datasets. A prototype of this application has been provided for data discovery in the cancer chemoprevention domain and various use case scenarios have been documented and provided. We ensured that the application was developed strictly using JavaScript and SVG for which all browsers have a native interpreter. The evaluation of the application proves that the usage of query elements in a user-proposed DSL supports quick identification in query formulations and data discovery and provides increased intuitiveness to the platform compared to various other applications in the same domain. The highly modular nature of this application, and the defined graphic rules, promise a plug-and-play architecture for future developers of this platform. As the application is independent of the underlying DSL, graphic rules
and the query engine, ReVeaLD can be implemented as a knowledge discovery system in various other domains.

ACKNOWLEDGEMENTS

The authors thankfully acknowledge the EU FP7 GRANATUM project, ref. FP7-ICT-2009-6-270139 and Science Foundation Ireland Lion 2. The authors would also like to acknowledge Claude Warren, for the development of the GRANATUM Query Engine, Panagiotis Hasapis, for the provision of sample use cases and scenarios in cancer chemoprevention knowledge discovery, and Ronan Fox, whose comments and advice were extremely valuable for the development of the platform, as well as for improving the manuscript. The authors would also like to acknowledge the anonymous evaluators of the platform.

REFERENCES


OpenLink iSPARQL. <http://dbpedia.org/isparql/> [accessed 09.05.13].


Appendix I – Evaluation SPARQL Queries

Usage Scenario I

SELECT DISTINCT * WHERE
{
  ?x0_ChemopreventiveAgent a granatum:ChemopreventiveAgent ;
  granatum:hasSource ?x2_Source ;
  granatum:referredIntoPublication ?x4_PublishedWork ;
  granatum:hasToxicity ?x3_Toxicity ;
  granatum:affectPathway ?x1_Pathway .

  ?x1_Pathway granatum:title ?x5_title .

  ?x2_Source granatum:commonName ?x6_commonName .

  ?x3_Toxicity granatum:level ?x7_level ;
  granatum:cellType ?x8_cellType ;
  granatum:species ?x9_species .

  ?x4_PublishedWork granatum:title ?x10_title
Usage Scenario II

```
SELECT DISTINCT * WHERE {

?x1_drugs a drugbank:drugs .
?x0_diseases a diseasome:diseases ;
   diseasome:possibleDrug ?x1_drugs ;
   rdfs:label ?x2_label .
?x1_drugs rdfs:label ?x4_label ;
   drugbank:chemicalFormula ?x5_chemicalFormula ;
   granatum:molecularWeight ?x3_molecularWeight ;
   drugbank:chemicalIupacName ?x7_chemicalIupacName ;
   drugbank:predictedLogpHydrophobicity ?x8_predictedLogpHydrophobicity ;
   drugbank:state ?x9_state ;
   drugbank:predictedWaterSolubility ?x6_predictedWaterSolubility

FILTER regex(xsd:string(?x2_label), "colon cancer", "is")
FILTER ( xsd:double(?x3_molecularWeight) < 1000 )
}
```

Usage Scenario III

```
INSERT DATA INTO <http://chem.deri.ie/granatum/>
{
    granatum:RNAMolecule rdf:type owl:Class ;
    rdfs:label "RNAMolecule"^^xsd:string ;
    rdfs:subClassOf :Molecule .
    granatum:containsNucleotide rdf:type owl:DatatypeProperty ;
    rdfs:domain :RNAMolecule ;
    rdfs:range xsd:string ;
    rdfs:label "contains nucleotide"^^xsd:string .
    granatum:DNAMolecule rdf:type owl:Class ;
    rdfs:label "DNAMolecule"^^xsd:string ;
    rdfs:subClassOf :Molecule .
}
```

Usage Scenario IV
SELECT * WHERE {
  ?x0_Strain a granatum:Study ;
    granatum:describedInPublication ?x1_PublishedWork ;
    granatum:containAssay ?x2_Assay .
  ?x1_PublishedWork granatum:title ?x4_title .
  ?x2_Assay granatum:hasInput ?x3_Molecule .
  ?x3_Molecule rdfs:label ?x6_label ; owl:sameAs ?x5_sameAs .
  FILTER regex(xsd:string(?x4_title), "mechanism-based in vitro screening", "is")}

Appendix II – ReVealD Usability Evaluation Questionnaire

Role as a GRANATUM ReVealD User:

<table>
<thead>
<tr>
<th>Question</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Are you able to find the relevant biomedical concept of your interest in the context of Cancer Chemoprevention e.g. Molecule, Drug, Chemopreventive Agent?</td>
<td>24</td>
<td>0</td>
</tr>
<tr>
<td>Are you able to easily add any new concept/literal property to the existing ones using the Model Incrementation tool?</td>
<td>16</td>
<td>8</td>
</tr>
<tr>
<td>Can you formulate the queries easily after watching the explanatory demo video under ‘Help’?</td>
<td>24</td>
<td>0</td>
</tr>
<tr>
<td>Are you able to launch the lens dialog window from within the data browser and retrieve more information about any specific entity?</td>
<td>19</td>
<td>5</td>
</tr>
<tr>
<td>Are you able to download the results or save them in the GRANATUM Platform?</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>Are the retrieved results useful to you in your in-silico modeling tools?</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>Can you search the relevant Literature Documents by their abstracts using ReVealD?</td>
<td>19</td>
<td>5</td>
</tr>
<tr>
<td>Question</td>
<td>General Opinion (Summarized)</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-----------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Are there any specific concepts/ literal properties that you think should be a part of the domain model but are missing?</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>What other functionalities you think should be a part of such a system to make it more effective and user-friendly?</td>
<td>Google-like Search Interface (NL-search)</td>
<td></td>
</tr>
<tr>
<td>What more features do you expect from ReVeaLD?</td>
<td>More Search Results</td>
<td></td>
</tr>
</tbody>
</table>